Final Exam *(Saturday 23-3-2013)* Time: 3 Hrs.

الإجابة النموذجية للامتحان النهائي

Q1 (13 pts)

a) Find the word or phrase from the table below that best matches the description. ***(4pts)***

a.	Cache		Small, fast memory that acts as a buffer for the main memory.
b.	ISA		Specific interface that the hardware provides the low-level software.
c.	Assembler		Program that converts a symbolic version of an instruction into the binary version.
d.	CPU		Component of the computer where all running programs and associated data reside.
e.	Instruction		Single software command to a processor.
f.	MIPS		Sometimes is used as a Performance metric.
g.	Miss Penalty		Is the time to replace block in upper level plus time to deliver data to the processor.
h.	Hit Time		Is the time to access the upper level of the memory hierarchy, which includes the time needed to determine whether the access is a hit or a miss.

b) What is the advantages of IEEE-754 standard for floating point numbers?. ***(3pts)***

- *Simplified presenting of floating-point numbers. Unified the algorithms of floating-point numbers. Increased the accuracy of floating-point numbers.*
- *Encoding of exponent and fraction simplifies comparison, integer comparator used to compare magnitude of FP numbers.*
- *Includes special exceptional values: NaN and ±∞. Special rules are used such as: 0/0 is NaN, sqrt(–1) is NaN, 1/0 is ∞, and 1/∞ is 0. Computation may continue in the face of exceptional conditions.*
- *Denormalized numbers to fill the gap between smallest normalized number 1.0 × 2^{Emin} and zero Denormalized numbers, values 0.F × 2^{Emin} , are closer to zero.*

c) Explain the replacement and write policies in the cache memory. ***(3pts)***

Replacement:

- *Random replacement : Candidate blocks are randomly selected. One counter for all sets (0 to m – 1): incremented on every cycle. On a cache miss replace block specified by counter.*

- *First In First Out (FIFO) replacement: Replace oldest block in set. One counter per set (0 to m – 1): specifies oldest block to replace. Counter is incremented on a cache miss.*
- *Least Recently Used (LRU): Replace block that has been unused for the longest time. Order blocks within a set from least to most recently used. Update ordering of blocks on each cache hit. With m blocks per set, there are m! possible permutations.*

Write:

- *Write through - write to memory, stall processor until done.*
- *Write back - delay write to memory until block is replaced in cache.*
- *Write buffer - place in buffer. Used in pipeline allows pipeline to continue.*

d) What is the difference between signed and unsigned arithmetic instructions in MIPS processor?. ***(3pts)***

The difference between signed and unsigned arithmetic instructions in MIPS processor is to control whether a trap is executed on overflow (Add instructions) or an overflow is ignored (Add unsigned instruction).

Q2 (17 pts)

a) The following steps are used to multiply two 5 bits signed numbers.
Which algorithm was used to perform this multiplication? ***(2pts)***
Complete the steps in the table and find the value of Multiplier, Multiplicand and Product in decimal? ***(7pts)***

1	Shift Right	
2	Shift Right	
3	Subtract	
	Shift Right	
4	Add	
	Shift Right	
5	Subtract	Result after subtraction 000011000010

The Algorithm is Booth's Algorithm for Signed Multiplication.

Works for two's complement numbers. Key idea: test 2 bits of multiplier at once.

10 - subtract (beginning of run of 1's)

01 - add (end of run of 1's)

00, 11 - do nothing (middle of run of 0's or 1's)

The number is signed and 5 bit

The result at step 5 after subtraction is 00011000010

Last step is shift right after subtraction : 00001100001

Product (PR) = 0000110000 = 48_{10}

Multiplier (MR):
In step 5 Subtract → 000011000010 .(10 = MR4 & MR3)

MR4	MR3	MR2	MR1	MR0
1	0	?	?	?

In step 4 Add (01) → (01 = MR3 & MR2)

MR4	MR3	MR2	MR1	MR0
1	0	1	?	?

In step 3 Subtract (10) → (10 = MR2 & MR1)

MR4	MR3	MR2	MR1	MR0
1	0	1	0	?

In step 2 Shift Right (00 or 11) → but because in step 3 is subtraction → the two bits should be 00 not 11 → (00 = MR1 & MR0)

MR4	MR3	MR2	MR1	MR0
1	0	1	0	0

MR = 10100 = -12_{10}
MC = PR/MR = 48/ –12= –4.

b) Calculate the following half precession floating point arithmetic operations. ***(8pts)***

Half Precession format (1bit for sign, 5 bits for Exponent, and 10 bits for Fraction)

i.

```
  0111111110000010
- 0000111011000000
```

0 11111 1110000010
S= 0
E=31
Fraction Non-zero
∴ This number is a special case (NaN)
NaN operation any number = NaN.

ii.

```
  0111111000000001
× 1000111011000000
```

0 11111 1000000001
S= 0
E=31
Fraction Non-zero
∴ This number is a special case (NaN)
NaN operation any number = NaN.

Q3 (20 pts)

a) Write a sequence of MIPS instructions which can discover if there is an overflow or not in signed addition. ***(4pts)***

```
add $t0,$t1,$t2                  # $t0 = sum.
xor $t3,$t1,$t2                  # Check if signs differ
slt $t3,$t3,$zero                # $t3 = 1 if signs differ
bne $t3,$zero,No_overflow        # $t1, $t2 signs so no overflow
xor $t3,$t0,$t1                  # signs =; sign of sum match too?
                                 # $t3 negative if sum sign different
slt $t3,$t3,$zero                # $t3 = 1 if sum sign different
bne $t3,$zero,Overflow           # All three signs ≠; go to overflow
```

b) Two different compilers are being tested on the same program for a 6 GHz machine with three different classes of instructions: Class A (branch instructions) , Class B (load instructions), and Class C (other instructions), which require 3, 5, and 1 cycles, respectively. The instructions produced by the first and the second compiler are shown below.

Compiler 1	Compiler 2
`lui $s0,0x100`	`lui $s0,0x100`
`ori $s0,$s0,0x80fc`	`ori $s0,$s0,0x80fc`
`ori $s6,$0,0`	`ori $s1,$0,255`
`ori $t0,$0,0`	`or $s2,$0,$0`
`next: lbu $s5,0($s0)`	`or $s6,$0,$0`
`add $s6,$s6,$s5`	`or $s4,$0,$0`
`addi $s0,$s0,1`	`ori $t0,$0,3`
`addi $t0,$t0,1`	`lw $s5,0($s0)`
`slti $t1,$t0,3`	`next: and $s3,$s5,$s1`

```
        bne  $t1,$0,next
        div  $s6,$t0
        mflo $s6
        mfhi $s7
```

```
        srlv $s3,$s3,$s4
        add  $s6,$s6,$s3
        addi $s4,$s4,8
        sll  $s1,$s1,8
        slti $t1,$s4,24
        bne  $t1,$0,next
        div  $s6,$t0
        mflo $s6
        mfhi $s7
```

The function of this program is to calculate the medium of 3 numbers, which are sorted as a big-endian in the memory.

ADDRESS	DATA
0×010080FC	0×19230DF7
0×01008100	0×F5CE67A3
0×01008104	0×112BC49A

i. What is the content of registers $s6 and $s7 in decimal after execution this program? ***(3pts)***.

The Content of $s6 and $s7:
The function of the program is to calculate medium of 3 numbers
Let us track the instructions produced by compiler 1
$s5 = 0x010080FC → the base address.
Memory byte order is big-endian → [0x010080FC] = 0x19
[0x010080FE] = 0x23
[0x010080FD] = 0xd
LBU instruction is used to load one byte from the memory to register $s5
$s6 initially =0
$s6 =$s6+$s5 = 0x23
next loop $s6= 0x23+0x19
next loop $s6= 0x23+0x19+0xd
Finally $s6/$t0 : $t3 at last loop = 3
The quotient will be in LO register and Remainder in HI Register
mflo $s6
Content of $s6 = 0x18 = 24_{10}
mfhi $s7
Content of $s7 = 0x1= 1

ii. Which compiler produces a better execution time? ***(8pts)***.
Clock Rate = 6 GHz

Class	*CPI*	*Instruction Count (IC) Compiler 1*	*Instruction Count (IC) Compiler 2*
Class A (Branch instructions)	*3*	*3*	*3*
Class B (Load instructions) تمت الإشارة في الإمتحان على أن هذاالكلاس خاص فقط ب *LBU and LW instructions*	*5*	*3*	*1*
Class C (Other instructions)	*1*	*7+4×3=19*	*10+6×3=28*
		Total IC =25	*Total IC = 32*

Complier 1:

CPU Cycles $_{Compiler\ 1}=\sum_{i=1}^{3} CPI_i\ IC_i=3\times3+5\times3+1\times19=$ *43 cycles*

CPU $_{ex\_time}$ *= CPU Cycles / Clock Rate = 43/6GHZ = 7.16 n sec*

Complier 2:

CPU Cycles $_{Compiler\ 2}=\sum_{i=1}^{3} CPI_i\ IC_i=3\times3+5\times1+1\times28=$ *42 cycles*

CPU $_{ex\_time}$ *= CPU Cycles / Clock Rate = 42/6GHZ = 7 n sec*

∴ Compiler C2 produces a better execution time.

iii. Which compiler produces a higher MIPS? ***(5pts)***.

MIPS = IC/(*CPU* $_{ex\_time}$ ** 10^6)*

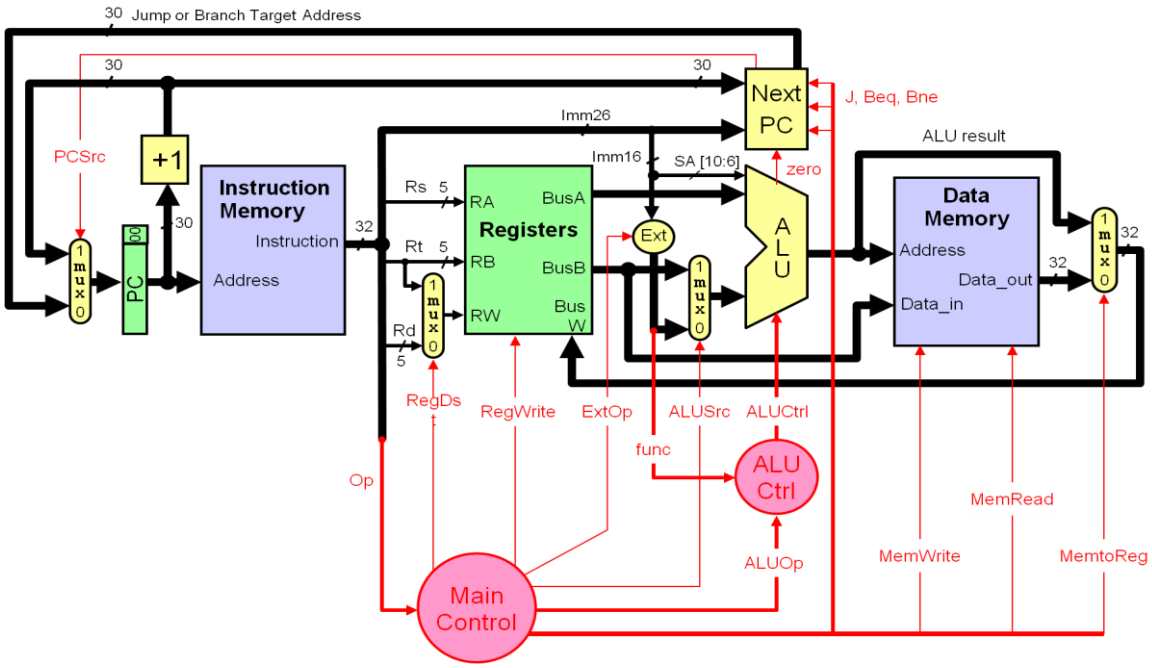
Compiler 1: MIPS = 3491.6 Million instruction per second

Compiler 2: MIPS = 4571.4 Million instruction per second

∴ Compiler 2 produces a higher MIPS.

Q4 (25 pts)

Consider the following single-cycle datapath for the MIPS processor implementing a subset of the instruction set (R-Type, Immediate Arithmetic and Logic, LW/SW, jump and branch instructions):



a) If ExtOp signal has a stuck at 0 or 1 faults. Which instructions mentioned above will not work correctly? Explain why.

(i) ExtOp stuck at 0. ***(5pts)***

Immediate Arithmetic (e.g. addi), LW and SW instructions will not work correctly. Because there are arithmetic operations so we have to extend the sign of the 16 bit immediate [15:0], (Sign Extended).

(ii) ExtOp stuck at 1. ***(5pts)***

Immediate Logic (e.g. ori) will not work correctly.
Because it is a logical operation so we don't need to extend the sign of the 16 bit immediate [15:0], (Zero Padding).

b) Suppose that the ALU has a shifter which is used for R-type shift instructions. And suppose that the content of $s0 = 0x010080CC. Explain how can you modify the datapath and control unit to implement (*lui, lb & lbu)* instruction. *Draw the modified datapath only and write all the control signals values as shown in the table below.* **`(lui rt, immediate), lb $s1,imm($s0)#(lb rt, imm(rs))`**
`lbu $s2, imm($s0) #(lbu rt, imm(rs)).` ***(15pts)***

تم رسمها في التيتوريا

Op	*PC Src*	*Reg Dst*	*Reg Write*	*Ext Op*	*ALU Src*	*ALU Op*	*Beq*	*Bne*	*J*	*Mem Read*	*Mem Write*	*Memto Reg*	*Additional Control Signals*		
													LUI	*EXT Op2*	*LB/LBU*
lui	*1*	*1*	*1*	*x*	*0*	*sll*	*0*	*0*	*0*	*0*	*0*	*1*	*1*	*x*	*0*
lb	*1*	*1*	*1*	*1*	*0*	*add*	*0*	*0*	*0*	*1*	*0*	*0*	*x*	*1*	*1*
lbu	*1*	*1*	*1*	*1*	*0*	*add*	*0*	*0*	*0*	*1*	*0*	*0*	*x*	*0*	*1*

c) Implement add, xori & beq instructions to the data-path. *Write all the control signals values as shown in the table below.* `(add rd,rs,rt),(xori rt,rs, imm), (beq rs,rt,label).` ***(10pts)***

Op	*PC Src*	*Reg Dst*	*Reg Write*	*Ext Op*	*ALU Src*	*ALU Op*	*Beq*	*Bne*	*J*	*Mem Read*	*Mem Write*	*Memto Reg*
add	*1*	*0*	*1*	*x*	*1*	*add*	*0*	*0*	*0*	*0*	*0*	*1*
xori	*1*	*1*	*1*	*0*	*0*	*xor*	*0*	*0*	*0*	*0*	*0*	*1*
beq	*0*	*x*	*0*	*x*	*1*	*sub*	*1*	*0*	*0*	*0*	*0*	*x*

للعلم تم تنبيه الطلبة على أن السؤال الرابع b إجباري والسؤالين a & c يختار أحداهما

Q5 *(25 pts)*

a) Consider a cache is organized as a direct-mapped with 2 bits offset, and total amount of bits in each row is 42 bits. Assume that the cache is initially empty.

i. Compute the total number of bits required to implement this cache. ***(3pts)***

b = 2 bits → block size = 2^b =2^2 = 4 bytes
*Total amount of bits in each row =42 = valid bit + tag bits + block size = 1+ tag bits + 4 * 8.*
42=1+tag bits + 32 → tag bits = 42-33 = 9 bits.
From the table reference address size is 16 bit -→ index bits = 16-9-2 =5 bits
Number of blocks = 2^n = 2^5 =32 block.

*Total Number of bits required to implement this cache = total number of bits in each row * number of blocks =42*32= 1344 bits = 168 byte.*

ii. What is the size of this cache according to the number of bytes stored in the cache? ***(3pts)***
Cache data size = 2^(n+b)= 2^(5+2)= 2^7 = 128 bytes.

iii. Determine whether if the addresses are hit or miss, and calculate the miss rate. ***(4pts)***
2099, 2110, 1971, 1599, 2111, 1968, 1598, 1982, 2097, 2111, 1982, 2098

16 bit Byte Address (Decimal)	Address (binary)	Tag Decimal	Index Decimal	Offset Decimal	Hit / Miss
2099	*0 0001 0000 01100 11*	*16*	*12*	*3*	*Miss*
2110	*0 0001 0000 01111 10*	*16*	*15*	*2*	*Miss*
1971	*0 0000 1111 01100 11*	*15*	*12*	*3*	*Miss*
1599	*0 0000 1100 01111 11*	*12*	*15*	*3*	*Miss*
2111	*0 0001 0000 01111 11*	*16*	*-15*	*3*	*Miss*
1968	*0 0000 1111 01100 00*	*15*	*12*	*0*	*Hit*
1598	*0 0000 1100 01111 10*	*12*	*15*	*2*	*Miss*
1982	*0 0000 1111 01111 10*	*15*	*15*	*2*	*Miss*
2097	*0 0001 0000 01100 01*	*16*	*12*	*1*	*Miss*
2111	*0 0001 0000 01111 11*	*16*	*15*	*3*	*Miss*
1982	*0 0000 1111 01111 10*	*15*	*15*	*2*	*Miss*
2098	*0 0001 0000 01100 10*	*16*	*12*	*2*	*Hit*
MISS Rate (%) = 10/12 = 83.3 %					

b) Consider a cache is organized as a fully associative with four comparators, and the cache data size is 32 byte. Calculate the miss rate for these reference addresses. Assume that the cache is initially empty and the replacement policy which used in this cache is FIFO. ***(15 pts)***
107, 126, 111, 86, 76, 70, 107, 74, 86, 107, 125, 86

Cache data size = 32 bytes = m * 2^ b = 4 * 2^b → b= 3 bits → tag = 16-3 =13 bits
The replacement policy is FIFO.

16 bit Byte Address Decimal	Address (binary)	Tag Decimal	Offset Decimal	Hit / Miss
107	1101 011	13	3	Miss
126	1111 110	15	6	Miss
111	1101 111	13	7	Hit
86	1010 110	10	6	Miss
76	1001 100	9	4	Miss
70	1000 110	8	6	Miss
107	1101 011	13	3	Miss
74	1001 010	9	2	Hit
86	1010 110	10	6	Hit
107	1101 011	13	3	Hit
125	1111 101	15	5	Miss
86	1010 110	10	6	Miss
MISS Rate (%) = 8/12 = 66.667 %				

c) Consider three processors with different cache configurations. *Cache 1:* Direct-mapped, cache data size 64 bytes with 16 blocks, Instruction miss rate is 3%; data miss rate is 5%. *Cache 2:* Direct-mapped with 3bits offset, Instruction miss rate is 2%; data miss rate is 6%. *Cache 3:* Two-way set associative, cache data size 64 bytes with 3bits index, Instruction miss rate is 2%; data miss rate is 3%. For these processors, 65% of the instructions contain a data reference. Assume that the cache miss penalty is 8 + block size in bytes. The CPI for this workload was measured on a processor with cache 1 and was found to be 2.3.

i. Determine which processor spends the most cycles on cache misses. ***(4pts)***

	Cache 1	*Cache 2*	*Cache 3*
Block size	*Direct Mapped* *Cache data 64 bytes.* *Number of block= 16→ 2^n =16 → n=4 bits* *64=2^(n+b) =2^(4+b)* *b=2 →Block Size = 2^2=4 bytes*	*Direct Mapped* *3bits offset → Block size = 2^b=8 bytes*	*2-way set associative* *Cache data 64 bytes.* *Index (n) = 3 bits* *64= m* 2^(n+b)* *64= 2* 2^(3+b)* *b = 2 -→* *Block Size = 2^2=4 bytes*
Miss penalty 8+ block zize	*8+4=12*	*8+8=16*	*8+4=12*
Memory Stall Cycles Per Instruction = I-Cache Miss Rate × Miss Penalty + LS Frequency × D-Cache Miss Rate × Miss Penalty *Combined Misses Per Instruction = I-Cache Miss Rate + LS Frequency × D-Cache Miss Rate* *Stall Cycles Per Instruction = Combined Misses Per Instruction × Miss Penalty* *65% of instructions contain data reference → LS frequency = 0.65*			
	Cache 1	*Cache 2*	*Cache 3*
Combined Misses Per Instruction	*0.03+0.65 * 0.05 = 0.0625*	*0.02+0.65 * 0.06 = 0.059*	*0.02+0.65 * 0.03 = 0.0395*
Memory Stall Cycles Per Instruction	*0.0625*12=0.75 stall cycles per instruction*	*0.059*16=0.944 stall cycles per instruction*	*0.0395*12=0.474 stall cycles per instruction*
∴ Cache 2 spends the most cycles on cache misses			

ii. If the cycle times for the first and third processors are 420 ps, and 310 ps for the second processor. Determine which processor is the fastest and which is the slowest. ***(4pts)***

CPU Time = I-Count × $CPI_{MemoryStalls}$ × Clock Cycle

$CPI_{MemoryStalls}$ = $CPI_{PerfectCache}$ + Mem Stalls per Instruction

$CPI_{MemoryStalls}$ for Cache 1 is given = 2.3 cycles per instruction

$CPI_{PerfectCache}$= $CPI_{MemoryStalls\ (Cache\ 1)}$ - Mem Stalls per Instruction $_{(Cache\ 1)}$

$CPI_{PerfectCache}$= 2.3 - 0.75 = 1.55 cycles per instruction

$CPI_{MemoryStalls\ (Cache\ 2)}$= 1.55+0.944=2.494 cycles per instruction

$CPI_{MemoryStalls\ (Cache\ 3)}$= 1.55+0.474=2.024 cycles per instruction

CPU Time = I-Count × $CPI_{MemoryStalls}$ × Clock Cycle

CPU Time $_{cache\ 1}$= IC × 2.3 × 420 = 966 IC ps

CPU Time $_{cache\ 2}$ = IC × 2.494 × 310 = 773.14 IC ps

CPU Time $_{cache\ 3}$ = IC × 2.024 × 420 = 850.08 IC ps

∴ The processor on cache 2 is the fastest and the processor on cache 3 is the slowest

iii. By how much the fastest processor is faster than the slowest one. ***(2pts)***

The fastest processor is faster than the slowest one by CPU Time $_{cache\ 3}$/ CPU Time $_{cache\ 2}$ = 1.25.

للعلم تم تنبيه الطلبة على أن السؤال الخامس b إجباري والسؤالين a & c يختار أحداهما

GOOD LUCK ☺